# HTML:Строчные и блочные тэги. Правила вложенности.

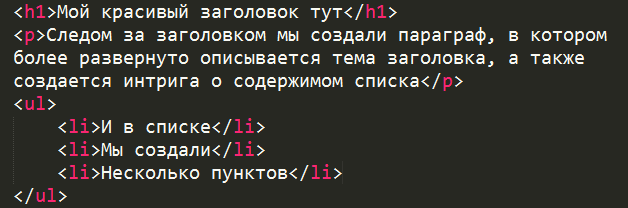
# CSS: Отступы.

## Понятие строки и блока

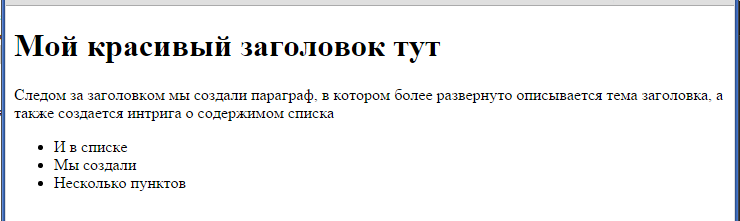
Мы с вами уже познакомились с html тэгами и узнали, что у них есть свое поведение в браузере - какие-то из тэгов изменяют размер шрифта, его цвет, наклонность, жирность (например, тэги h1, a, i, b, em, strong), а некоторые вроде как не добавляют никаких свойств шрифту (тэги p, span, div). Так мы узнали, что не каждый тэг изменяет вложенный в него текст, но у каждого тэга есть набор определенных свойств.

Одним из ключевых понятий в верстке является различение блока и строки. Пока не очень ясно, о чем речь, поэтому давайте посмотрим на примерах: что будет, если поставить в странице подряд несколько блоков?

Например, в разметке мы создадим заголовок, за ним параграф, за которым последует список.

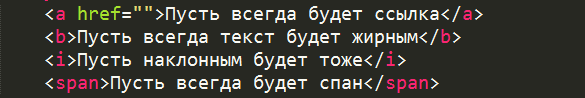


Если мы откроем этот документ в браузере, то увидим, что каждый элемент расположился на **новой строке, под** предыдущим блоком

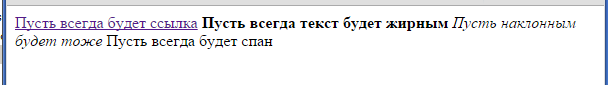


Элементы с таким поведением мы будем называть **блочными.** Блочные элементы занимают **всю ширину экрана**, даже если в них совсем немного содержимого. В странице блочные элементы располагаются **один под другим,** начиная от верха страницы.

Давайте теперь создадим разметку с другими тэгами, такими как span, a, b, i, em или strong.



Мы также, как и в предыдущем примере поставили тэги один за другим, в редакторе даже поставили их каждый с новой строки. Но в браузере они расположились на **одной строке, подряд друг за другом слева направо.**



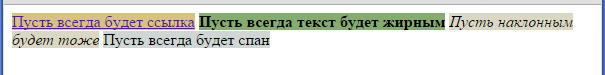
Элементы с похожим поведением мы будем называть **строчными**. Строчные элементы занимают на экране пространство **по количеству содержимого**. Чем больше текста - тем больше размер элемента. В странице строчые элементы располагаются **справа** от предыдущего и переходят на новую строку только если содержимое не вмещается в одну строку.

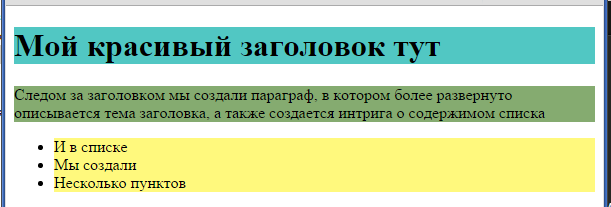
Итак, на текущий момент мы выяснили, что между строками и блоками есть различия:

|  |  |
| --- | --- |
| **Блоки** | **Строки** |
| Располагаются по вертикали друг под другом. | Располагаются по горизонтали слева направо |
| Ширина во весь экран | Ширина равна содержимому |
| Всегда начинаются с новой строки | Переходит на новую строку, если содержиое не вместилось в одну |

Если заглянуть в логику браузера, то мы увидим, что вокруг каждого тэга он нарисует прямоугольник.

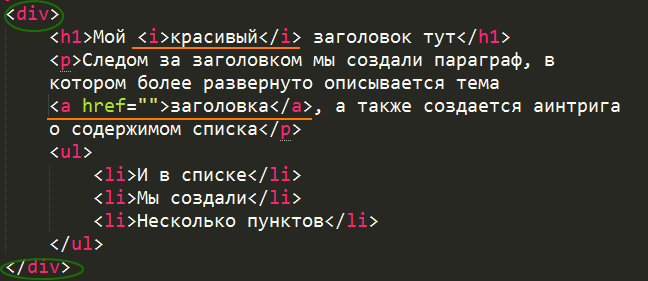
Если элемент строчный, то размер прямоугольника будет равен содержимому тэга



Если элемент блочный, то прямоугольник будет занимать всю доступную ширину 

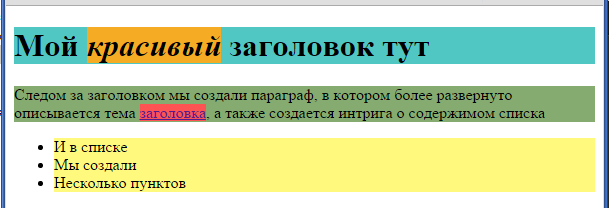
Итак, для каждого тэга браузер создает отдельный прямоугольник, этакую коробку, в которую вкладыват текст-содержимое. Если браузер встретился с блочным элементом, он берет самую широкую коробку и складывает туда наш текст. Тексту свободно и просторно с большой блочной коробке. Каждую коробку он ставит на предыдущую, в стопку. Если браузер встретился со строчным элементом, то он жадничает, и выбирает коробку ровно по размеру текста. Текст плотно упакован в своей строчной коробке, лишнего пространства ему браузер не выделил. Строчные коробки он ставит рядом друг с другом, на одну полочку. Вот такой он, браузер-упаковщик-содержимого-по-коробкам.

Кроме подбора коробок, наш браузер умеет экономить пространство и вкладывать коробки друг в друга. Например, мы можем положить ссылку в параграф, часть заголовка сделать наклонной при помощи тэга, а все содержимое страницы положить в одну очень большую коробку.



Мы обернули все содержимое в блок div (самая большая коробка), в заголовок положили строчный тэг <i>, а внутрь параграфа вложили ссылку <a>

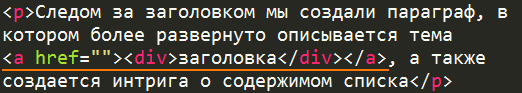
Браузер увидит страницу вот так (конечно, без этих умопомрачительно красиво подобранных цветов, их мы сделали для демонстрации):



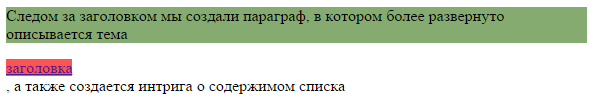
Браузер расположил строчные элементы внутри блочных, но каждый строчный элемент все равно вложен в свою маленькую тесную коробочку.

Так мы выяснили, что браузер легко вкладывает строчные тэги в блочные, маленькие коробки в большие.

Нам теперь интересно, а что будет, если попытаться вложить блочный элемент в строчный? Как браузеру удастся впихнуть большую коробку в маленькую? Попробуем положить в нашу строчную ссылку блочный элемент, напрмер, див?



В браузере все сломалось:



Мы можем конечно приложить усилия и стилизовать даже такой неправильный вариант верстки и он будет красиво смотреться. Но так делать не нужно!!! это неправильно, не соответствует стандарту и т. д.

Мы вложили блочный тэг в строчный только один раз, в этом примере. Больше мы так делать с вами не будем никогда, потому что мы запомнили, что это не правильно!!!

Мы будем бережно относиться к работе нашего браузера-упаковщика и не будем заставлять его выполнять неправильные задачи, т.е. не будем вкладывать блоки в строки.

### Что еще нужно знать о вложенности?

Нельзя вкладывать:

* блоки в строки
* ссылки друг в друга
* параграфы друг в друга
* другие блоки (div, ul, li) в параграфы

Можно вкладывать

* строки в блоки
* строки в строки
* параграфы в блоки

## Отступы в CSS

В этом занятии мы с вами увидели, что браузер интерпретирует (видит, воспринимает) тэги, как прямоугольники разного размера. Но пока нам показалось, что он вокруг каждого тга рисует только один прямоугольник, не так ли? На самомо деле, он рисует целых 4(!), да-да, четыре прямоугольника вокруг каждого тэга. Зачем же ему так много лишней работы?

Давайте сначала посмотрим, что же за прямогольники он рисует:



На самом деле, наш упаковщик-браузер вкладывает содержимое тэга стразу в несколько коробок. В этом случае он ведет себя, как Кащей Бессмертный, только вместо иголки браузер упаковывает контент, содержимое нашего тэга.

Итак, контент браузер должен вложить в три коробки: контент в паддинг (вместо яйца), паддинг в бордер (вместо утки), бордер в марджин (вместо зайца).

И вот тут мы спросим браузер: не слишком ли ты много общаешься с темными личностями вроде Кащея? Не повредило ли это твоей светлой голове?

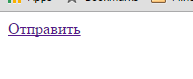
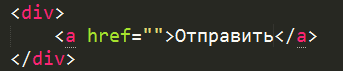
Но малыш браузер умный парень, он знает, что делает. Он вкладывает контент каждого тэга в коробки, которые помогут нам расположить блоки относительно друг друга. Давайте посмотрим, как дружище-браузер позаботился о счастье верстальщика.



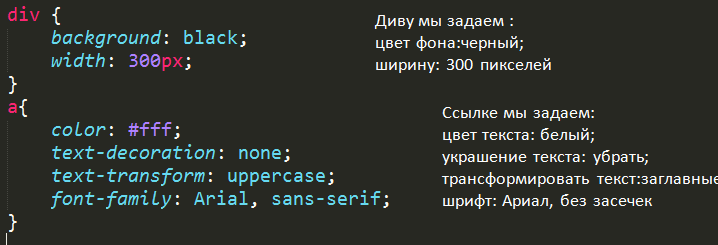
Допустим, нам нужно создать кнопку, прямоугольную, с белой границей и текстом, расположенным по центру кнопки. Кроме того, кнопка должна сама располагаться по центру черного блока.

В html создаем разметку, основой для кнопки нам послужит ссылка, вложенная в див:

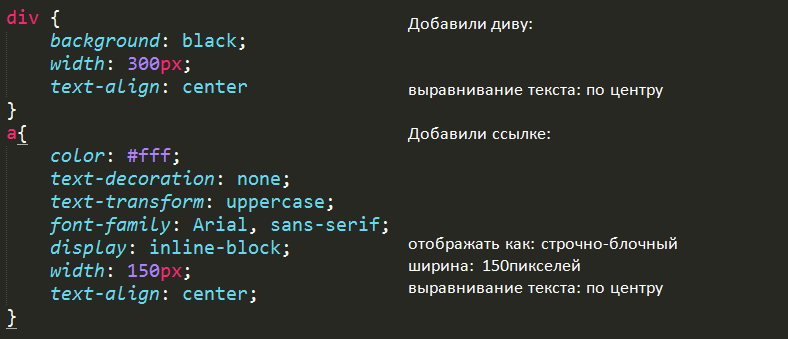
Вот что получилось в редакторе (слева) и в браузере (справа)



Пока в браузере не очень похоже, правда? Давайте добавим немного стилей для дива и для ссылки

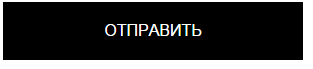


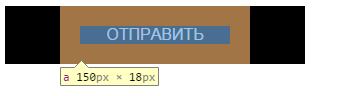
В браузере уже получилась что-то, больше похожее на нашу кнопку. Ей мы можем добавить еще стилей и выровнять текст по центру. кроме того, мы зададим ей ширину. Так как ссылка - это строчный элемент, ширина, высота, отступы на нее не действуют, поэтому мы изменим ее отображение на строчно-блочное, и тогда на нее будут действовать и свойства текста, и свойства блоков, например, мы сможем применить свойства ширины(свойство для блоков) 





Текст кнопки всал по центру блока, но ей все еще не хватает **границы и отступов.**

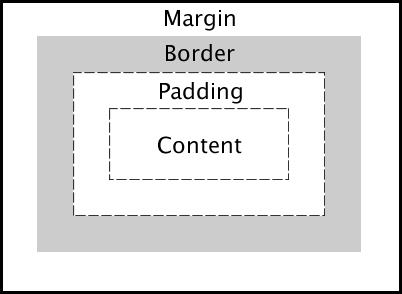
Вот тут-то мы и воспользуемся коробками, припасенными браузером для нашей кнопки. Добавим ей свойство margin, например: запишем в стилях margin:20px. И вот, как по волшебству, кнопка отодвинулась от границ родительского блока. 

Если посмотреть ее в инспекторе, то видно, что кнопка (подсвечена голубым) имеет размер 150px в ширину (мы задали ее в стилях), текст расположен по центру, а вокруг кнопки с появился внешний отступ (margin) размером в 20 пикселей. Добавьте кнопке еще свойство border: 2px solid #fff (граница: 2 пикселя сплошная белая) и полюбуйтесь результатом:

кнопка почти как на дизайне! Сейчас только граница расположена слишком близко к тексту, и нам бы добавить еще один отступ от текста до границы. Спасибо браузеру, он припас для нас паддинг - добавим в стилях свойство padding:10px;



Тройное УРА! Кнопка получилась совсем как на дизайне! Мы радуемся, но понимаем, что мы не за что бы не справились так быстро и просто, если бы не наш друг, который заранее положил коробку с кнопкой в три обертки: margin, border и padding.



Теперь давайте запомним все три замечательные свойства:

**Паддинг** - самый ближайший к содержимому отступ. Отступ от содержимого до границы блока

**Бордер** - граница блока.

**Марджин** - оступ от границы блока до бругих блоков.